Semantic Smoothing for Model-based Document Clustering

Xiaodan Zhang, Xiaohua Zhou, Xiaohua Hu
College of Information Science & Technology, Drexel University
xzhang@ischool.drexel.edu, xiaohua.zhou@drexel.edu, thu@ischool.drexel.edu

Abstract

A document is often full of class-independent “general” words and short of class-specific “core” words, which leads to the difficulty of document clustering. We argue that both problems will be relieved after suitable smoothing of document models in agglomerative approaches and of cluster models in partitional approaches, and hence improve clustering quality. To the best of our knowledge, most model-based clustering approaches use Laplacian smoothing to prevent zero probability while most similarity-based approaches employ the heuristic TF*IDF scheme to discount the effect of “general” words. Inspired by a series of statistical translation language model for text retrieval, we propose in this paper a novel smoothing method referred to as context-sensitive semantic smoothing for document clustering purpose. The comparative experiment on three datasets shows that semantic smoothing in conjunction with model-based clustering approaches is effective in improving cluster quality.

1. Introduction

Document clustering algorithms can be categorized into agglomerative and partitional approaches according to the underlying clustering strategy [6]. The agglomerative approaches initially assign each document into its own cluster and repeatedly merge pairs of clusters with shortest distance until only one cluster is left. The partitional approaches iteratively re-estimate the cluster generative model (or calculate the cluster centroid) and reassign each document into the closest cluster until no document is moved any longer. The clustering result of the agglomerative approach is free of the initialization and gives very intuitive explanation of why a set of documents are grouped together. However, in comparison with partitional approaches, it suffers from the $O(n^2)$ clustering time and performs poorly in general [12].

Steinbach et al. [12] argue that the agglomerative hierarchical clustering perform poorly because the nearest neighbors of a document belong to different classes in many cases. According to their examination on the data, each class has a “core” vocabulary of words and remaining “general” words may have similar distributions on different classes. Thus, two documents from different classes may share many general words (e.g. stop words) and will be viewed similar in terms of vector cosine similarity. To solve this problem, we should “discount” general words and “emphasize” more importance on core words in a vector. Besides, we think the poor performance of the agglomerative clustering can also be attributed to the sparsity of core words in a document. A document is often short and contains very few number of core words. Thus, two documents from the same class may share few core words and be falsely grouped into different clusters when using vector cosine similarity metric. To solve this problem, we should assign reasonable positive counts to “unseen” core words if its related topical words occur in the document.

Recent advances in document clustering have shown that model-based partitional clustering approaches are more efficient and effective than agglomerative clustering approaches in general [18]. However, two identified problems, the density of class-independent general words and the sparsity of class-specific core words, are also with the model-based approaches. Model-based partitional approaches estimate cluster models instead of document models. A cluster often contains much more than one document. Thus, the data sparsity problem is not as serious as in pairwise document similarity calculation. But if the size of the dataset for clustering is small or the dataset is extremely skewed up on different classes, the sparsity of core words will still be a serious problem. Besides, no matter how many documents a cluster have, general words always dominate the cluster; thus, discounting the effect of general words is always helpful to improve cluster quality.

Discounting seen words and assigning reasonable counts to unseen words are two exact goals of the probabilistic language model smoothing. To the best of our knowledge, the effect of model smoothing has not been extensively studied in the context of document clustering. Most model-based clustering
approaches simply use Laplacian smoothing to prevent zero probability [9] [10] [18] while most similarity-based clustering approaches employ the heuristic TF*IDF scheme to discount the effect of “general” words [12]. In contrast, the study of language model smoothing has been a hot topic in the community of information retrieval (IR) with the increasing popularity of the language modeling approach to IR in recent years [2] [15] [16] [19]. In this paper, we will adapt the smoothing techniques used in IR to the context of document clustering and hypothesize that the document or cluster model smoothing can significantly improve the quality of document clustering.

In IR, a simple but effective smoothing strategy is to interpolate document models with a background collection model. For example, Jelinek-Mercer, Dirichlet, Absolute discount [15] and Two-stage smoothing [16] are all based on this strategy. In document clustering, TF*IDF score is often used for document vectors. The effect of TF*IDF scheme is roughly equivalent to the effect of background model smoothing. However, a potentially more significant and effective smoothing method is what may be referred to as semantic smoothing where context and sense information are incorporated into the model [8]. The first trial of semantic smoothing may be dated back to latent semantic indexing (LSI) [3] which projects documents in the corpus into a reduced space where document semantics becomes clear. LSI explores the structure of term co-occurrence and can solve synonym problem very well. However, it brings noise while reducing the dimensionality because it is unable to recognize the polysemy of a same term in different context. In practice, LSI is also criticized for lack of scalability and interpretability. The recent integration of text semantics into document clustering may be the COBRA [13, 14] which uses the hierarchy of MeSH to measure the similarity of two biological terms and further two Medline abstracts. This approach needs domain ontology and thus is difficult to be applied to many public domains. Also, the relationship of two terms in ontology is static and will not be updated with the change of the corpus content.

Berger and Lafferty [2] proposed a kind of semantic smoothing approach referred to as the statistical translation language model which statistically mapped document terms onto query terms. With term translations, a document containing “star” may be returned for the query “movie” because two words have strong association on the topic of entertainment. Likewise, a document with the dimension of “star” but not “movie” may be merged into a cluster of “entertainment” together with a document containing “movie” but not “star”. However, like the LSI, this approach also suffers from the context-insensitivity problem, i.e., it is unable to incorporate contextual and sense information into the model. Thus, the resulting translation may be fairly general and contain mixed topics. For example, the word “star” can be either from the topic of “entertainment” (movie star) or from the topic of “military” (star war).

To overcome the context-insensitivity problem, we propose a novel context-sensitive semantic smoothing method suitable for document clustering, inspired by the topic signature language model [19]. The basic idea of the new smoothing method is to identify multiword phrases and then statistically map multiword phrases into individual document terms. Here, a multiword phrase can be viewed as a sub-topic or a latent dimension in LSI. However, the semantics of a phrase is clear and explicit since a multiword phrase is unambiguous in most cases. Thus, the translation of phrases to individual terms will be very specific. In addition, unlike the distribution of individual terms, the distribution of most multiword phrases depends on the topic. Therefore, phrases are very helpful to cluster documents.

We evaluate our semantic smoothing method in conjunction with a model-based agglomerative algorithm and a model-based K-Means algorithm on three datasets: 20-newsgroups, TDT2, and LA Times. The experiment shows that the agglomerative approach with semantic smoothing of document models significantly outperform with traditional vector cosine similarity measure and the model-based K-Means with semantic smoothing consistently achieves better results than with simple background smoothing and Laplacian smoothing when the dataset is small. The rest of the paper is organized as follows: Section 2 describes the semantic smoothing method. Section 3 shows the clustering methods. In section 4, we present and discuss experiment results. Section 5 concludes the paper.

2. Semantic smoothing

2.1. Document model smoothing

Berger and Lafferty [2] proposed a statistical translation language model for information retrieval. It smooths a document model by statistically mapping document terms to query terms. That is,

\[
p(q | d) = \sum_w t(q | w)p(w | d)
\]
where \( p(q|w) \) is the probability of translating the document term \( w \) to the query term \( q \) and \( p(w|d) \) is the maximum likelihood estimator of the document model. This model significantly improves the retrieval performance over the simple language model [2]. However, it is unable to incorporate contextual and sense information into the model. Thus, the resulting translation may be fairly general and contain mixed topics. To overcome this problem, Zhou et al. [19] proposed a context-sensitive semantic smoothing framework referred to as topic signature language model. The basic idea of the framework is to decompose a document into a set of context-sensitive topic signatures (e.g., concept pairs or multiword phrases) and then to translate topic signatures into individual terms. Because those topic signatures are unambiguous in most cases, the resulting transition will be very specific.

The topic signature language model is described in equation (2). It is a mixture model with two components: a simple language model and a topic signature translation model.

\[
p_{\lambda}(w|d) = (1-\lambda)p_{\lambda}(w|d) + \lambda p_{\lambda}(w|d)
\]

(2)

The translation coefficient \( \lambda \) is to control the influence of the translation component \( p_{\lambda}(w|d) \) in the mixture model. With some training data, the translation coefficient can be optimized by maximizing certain clustering quality metrics such as entropy [12] and NMI [1].

The simple language model in (2) can be easily obtained using the maximum likelihood estimator (MLE) together with some background smoothing techniques such as Jelinek-Mercer, dirichlet, absolute discount [15] and two-stage language model [16]. In this paper, we take the Jelinek-Mercer smoothing. That is,

\[
p_{\lambda}(w|d) = (1-\alpha)p_{\lambda}(w|d) + \alpha p(w|C)
\]

(3)

where \( \alpha \) is a coefficient accounting for the background collection model \( p(w|C) \) and \( p_{\lambda}(w|d) \) is the MLE document model. In our experiment, \( \alpha \) is set to 0.5.

The topic signature translation model smoothes document models by statistically mapping context-sensitive topic signatures onto individual terms. It has the following generic format:

\[
p_{\lambda}(w|d) = \sum_{k} p(w|\theta_{k}) p_{\lambda}(t_{k}|d)
\]

(4)

Here \( t_{k} \) denotes topic signatures identified in a document. The probability of translating \( t_{k} \) to individual term \( w \) is estimated with the approach described in Section 2.2. The likelihood of a given document generating the topic signature \( t_{k} \) can be estimated with

\[
p_{\lambda}(t_{k}|d) = \frac{c(t_{k},d)}{\sum_{t} c(t_{k},d)}
\]

(5)

where \( c(t_{k},d) \) is the frequency of the topic signature \( t_{k} \) in a given document \( d \).

The topic signature model has the advantage that its translation result will be very specific. However, not all topics in a document can be expressed by topic signatures such as concept pairs and multiword phrases. We have to interpolate the topic signature translation model with a simple language model in order to capture the missing points while the translation model proposed by Berger and Lafferty [2] does not have to do so.

Figure 1. Illustration of document indexing. \( V_{t} \), \( V_{d} \) and \( V_{w} \) are phrase set, document set and word set, respectively.

2.2. Phrase extraction and translation

Zhou et al. [19] implemented topic signatures as concept pairs and developed an ontology-based approach to extract concepts and concept pairs from documents. However, for many domains, ontology is not available. For this reason, we propose the use of multiword phrases as topic signatures and employ Xtract [11] to identify phrases in documents. Xtract is a kind of statistical extraction tool with some syntactic constraints. It is able to extract noun phrases frequently occurring in the corpus without any external knowledge. Xtract uses four parameters, strength \( k_{0} \), peak z-score \( k_{1} \), spread \( U_{0} \), and percentage frequency \( T \), to control the quantity and quality of the extracted phrases. In the experiment, the four parameters are set to 1, 1, 4, and 0.75, respectively.

We index all documents in a given collection \( C \) with terms (individual words) and topic signatures (phrases) as illustrated in Figure 1. For each phrase \( t_{k} \), we have a set of documents \( D_{t} \) containing that phrase. Intuitively, we can use the document set \( D_{t} \) to estimate the translation model for \( t_{k} \), i.e., determining
the probability of translating the given phrase to terms in the vocabulary. If all terms appearing in the document set center on the topic signature $t_k$, we can simply use maximum likelihood estimator and the problem is as simple as term frequency counting. However, some terms address the issue of other topic signatures while some are background terms of the collection. We then use a mixture language model to remove the noise. Assuming the set of documents containing $t_k$ is generated by a mixture language model (i.e., all terms in the document set are either translated by the given topic signature model $p(w|\theta_k)$ or generated by the background collection model $p(w|C)$, we have:

$$p(w|D_k) = (1-\alpha) p(w|\theta_k) + \alpha p(w|C)$$

(6)

where $\alpha$ is a coefficient accounting for the background noise and $\theta_k$ denotes the translation model parameters.

Under this simple mixture language model, the likelihood of generating the document set $D_k$ is:

$$\log p(D_k|\theta, C) = \sum_w c(w, D_k) \log p(w|D_k)$$

(7)

where $c(w, D_k)$ is the document frequency of term $w$ in $D_k$, i.e., the cooccurrence count of $w$ and $t_k$ in the whole collection. The translation model can be estimated using the EM algorithm [4]. The EM update formulas are:

$$\hat{p}^{(n)}(w) = \frac{(1-\alpha)p^{(n)}(w|\theta_k)}{(1-\alpha)p^{(n)}(w|\theta_k) + \alpha p(w|C)}$$

(8)

$$p^{(n+1)}(w|\theta) = \frac{c(w, D_k) \hat{p}^{(n)}(w)}{\sum_i c(w, D_i) \hat{p}^{(n)}(w_i)}$$

(9)

In the experiment, we set the background coefficient $\alpha=0.5$. We also truncate terms with extreme small translation probabilities for two purposes. First, with smaller number of translation space, the document smoothing will be much more efficient. Second, we assume most terms with extreme small probability are noise (i.e. not semantically related to the given topic signature). In detail, we disregard all terms with translation probability less than 0.001 and renormalize the translation probabilities of the remaining terms.

Table 1. Examples of phrase-word translations. The three phrases are automatically extracted from the collection of 20-newsgroups by Xtract. We list the top 10 topical words for each phrase.

<table>
<thead>
<tr>
<th>Arab Country</th>
<th>Nuclear Power</th>
<th>Gay People</th>
</tr>
</thead>
<tbody>
<tr>
<td>Term</td>
<td>Term</td>
<td>Term</td>
</tr>
<tr>
<td>Arab</td>
<td>0.061</td>
<td>nuclear</td>
</tr>
<tr>
<td>country</td>
<td>0.048</td>
<td>power</td>
</tr>
</tbody>
</table>

3. Clustering methods

3.1. Model-based agglomerative clustering

The agglomerative approaches initially assign each document into its own cluster and repeatedly merge pairs of clusters with shortest distance till only the desired number of clusters is left. The generalized algorithm of agglomerative hierarchical clustering is shown in Figure 2. The distance measure between two clusters can be implemented in many ways including single linkage, complete linkage, and average linkage. In general, the single linkage suffers from the chaining problem, the complete linkage favors outliers, and the average linkage stands in the middle.

Table 2 The algorithm of agglomerative hierarchical clustering.

| Algorithm: Agglomerative Hierarchical clustering |
| Input: $D = \{d_1, \ldots, d_n\}$ and number of clusters $K$ |
| Output: clustered data samples given by the cluster identity vector $Y = \{y_1, \ldots, y_n\}, \ y_i \in \{c_1, \ldots, c_K\}$ |
| Steps: |
| 1. Treat each data sample as a unique cluster. |
| 2. Re-compute pairwise cluster distance. |
| 3. Merge most similar two clusters into one cluster |
| 4. Repeat step 2 and 3 till only $K$ clusters are left. |

Figure 2 The algorithm of agglomerative hierarchical clustering.

In our experiment, both single linkage and average linkage suffer severe chaining problem on all three testing datasets when using standard vector cosine as document similarity measure. For fair comparison, we use the complete linkage as cluster distance measure because it does not has the chaining problem on the testing dataset working with the baseline document similarity measure. With complete linkage criterion, the distance of two clusters is defined as the maximum
distance between one document in the first cluster and the other in the second cluster. That is,
\[ \Delta(c_i, c_j) \equiv \max \{ \text{dist}(d_m, d_n), d_m \in c_i, d_n \in c_j \} \] (10)
where \( \text{dist}(d_m, d_n) \) is the pairwise document distance measure. As most of previous approaches take standard vector cosine as the document similarity measure, we will take it as a baseline measure in the experiment.

As we stated in the section of introduction, the vector cosine measure has serious problem when measuring pairwise document distance for clustering purpose. Instead, we will estimate a language model for each document in the corpus with context-sensitive semantic smoothing (see equation 2). After that, we propose the use of the Kullback-Leibler divergence \[ \text{KL-divergence distance of } p(w|d_1) \text{ to } p(w|d_2) \text{ is defined as:} \]
\[
\Delta(d_1, d_2) = \sum_{w \in V} p(w \mid d_1) \log \frac{p(w \mid d_1)}{p(w \mid d_2)}
\] where \( V \) is the vocabulary of the corpus. The KL-divergence distance will be a non-negative score. It gets the zero value if and only if two document models are exactly same. However, KL-divergence is not a symmetric metric. Thus, we define the distance of two documents as the minimum of two KL-divergence distances. That is,
\[ \text{dist}(d_1, d_2) = \min \{ \Delta(d_1, d_2), \Delta(d_2, d_1) \} \] (12)

The calculation of KL-divergence involves scanning the whole vocabulary, which makes the solution intractable at al. To solve this problem, we truncate terms with its distribution probability less than 0.001 while estimating document model using the equation (2) and renormalize the probabilities of remaining terms. Because we keep terms with high probability values in document models, it makes almost no difference in clustering results.

### 3.2. Model-based partitional clustering

The model-based partitional clustering is a generalized version of the standard k-means [18]. It assumes that there are \( k \) parameterized models, one for each cluster. Basically, the algorithm iterates between a model re-estimation step and a sample re-assignment step as shown in Figure 3.

#### Algorithm: Model-based K-Means

**Input:** dataset \( D = \{d_1, ..., d_n\} \), and the desired number of clusters \( k \).

**Output:** trained cluster models \( \Lambda = \{\lambda_1, ..., \lambda_k\} \) and the document assignment \( Y = \{y_1, ..., y_n\} \), \( y_i \in \{1,...,k\} \).

**Steps:**
1. Initialize document assignment \( Y \).
2. Model re-estimation: \( \lambda_i = \arg \max_{\lambda} \sum_{d \in Y_i} \log p(d \mid \lambda) \)
3. Sample re-assignment: \( y_i = \arg \max_{\lambda} \log p(d_i \mid \lambda) \)
4. Stop if \( Y \) does not change, otherwise go to step 2

**Figure 3** Model based k-means algorithm [18]

The implementation of cluster model estimation depends on the word distribution assumption one made on the dataset. Zhong and Ghosh [18] compared several common generative models for document clustering and found out that the multinomial model consistently outperformed the multivariate Bernoulli model. For this reason, we choose multinomial model for evaluation. Based on the naive Bayes assumption, the log likelihood of document \( d \) generated by the \( j \)-th multinomial cluster model will be:
\[
\log p(d \mid c_j) = \sum_{w \in V} c(w,d) \log p(w \mid c_j)
\] (13)
where \( c(w,d) \) denotes the frequency count of word \( w \) in document \( d \) and \( V \) denotes the vocabulary. Thus, the problem remains to estimate parameters \( p(w \mid c_j) \) for the cluster model.

The parameter estimation of multinomial models is as simple as counting word frequency in the cluster [18]. However, one has to smooth the model in order to prevent zero probability caused by data sparsity. Laplacian smoothing approach is frequently used for model smoothing [18]. With Laplacian smoothing, we have,
\[
p(w \mid c_j) = \frac{1 + c(w,c_j)}{|V| + \sum_w c(w,c_j)}
\] (14)
where \( c(w,c_j) \) is the frequency count of word \( w \) in the \( j \)-th cluster. Obviously, Laplacian smoothing assigns all unseen words of a given cluster a fixed probability.

As we argued in the section of introduction, the purpose of model smoothing is more than preventing zero probability. Suitable smoothing approach will “discount” general words in the cluster and assign a reasonable probability to unseen class-specific “core” words. Thus, it can significantly improve the quality of clustering, especially when the size of document set for clustering is small. For this reason, we use the
semantic smoothing approach introduced in Section 2 to smooth the cluster model. Section 2 describes the details of document model estimation with semantic smoothing. The cluster model smoothing is very similar. That is,

\[ p(w|c_j) = (1-\lambda)p_{b}(w|c_j) + \lambda p_{t}(w|c_j) \]  

(15)

The translation coefficient \( \lambda \) is to control the influence of two components: a simple model \( p_{b}(w|c_j) \) and a translation model \( p_{t}(w|c_j) \) in the mixture model. The simple model is the mixture of the maximum likelihood model and the corpus model.

\[ p_{b}(w|c_j) = (1-\alpha)p_{ml}(w|c_j) + \alpha p(w|C) \]  

(16)

Here, we set \( \alpha \) to 0.5. The topic signature translation model is estimated with formula below:

\[ p_{t}(w|c_j) = \sum_{k} p(w|\theta_{t_k}) p(t_k|c_j) \]  

(17)

Here, \( t_k \) denotes the topic signature (phrase) and more details about the translation model are available in Section 2.

In our experiment, we compare semantic smoothing with background smoothing and Laplacian smoothing using model-based K-Means clustering. Furthermore, in order to compare our algorithm with basic partitional clustering approaches, we also implement a basic K-Means [5] using cosine similarity metric on three vector representation schemes. The calculation of the cluster centroid uses the following formula:

\[ centroid = \frac{1}{|C|} \sum_{d \in C} d \]  

(18)

The three vector representation schemes are: term frequency (TF), normalized term frequency (NTF: term frequency divided by the vector length), and TF*IDF (product of term frequency and inverse document frequency).

4. Experiment setting and result analysis

4.1. Evaluation methodology

Cluster quality is evaluated by three extrinsic measures, purity [17], entropy [12], and normalized mutual information (NMI) [1]. Here, we only list the result of NMI, an increasingly popular measure of cluster quality. The other three measures are consistent with NMI on all runs. NMI is defined as the mutual information between the cluster assignments and a pre-existing labeling of the dataset normalized by the arithmetic mean of the maximum possible entropies of the empirical marginals, i.e.,

\[ NMI(X, Y) = \frac{I(X; Y)}{(\log k + \log c) / 2} \]  

(19)

where \( X \) is a random variable for cluster assignments, \( Y \) is a random variable for the pre-existing labels on the same data, \( k \) is the number of clusters, and \( c \) is the number of pre-existing classes. Regarding the details of computing \( I(X; Y) \), please refer to [1]. NMI ranges from 0 to 1. The bigger the NMI is the higher quality the clustering is. NMI is better than other common extrinsic measures such as purity and entropy in the sense that it does not necessarily increase when the number of clusters increases.

We run agglomerative hierarchical clustering with complete linkage criterion. The two document distance metrics used for the comparative experiment are the traditional vector cosine and the Kullback-Leibler divergence proposed in this paper. For cosine similarity, we try two different representation schemes: term frequency and TF-IDF. For KL-divergence metric, we test eleven translation coefficients (\( \lambda \)) ranging from 0 to 1. When \( \lambda=0 \), it is actually the simple background smoothing.

We take the model-based k-means and the standard k-means for partitional clustering. For model-based k-means, we compare the effectiveness of three different smoothing methods: Laplacian smoothing, background smoothing, and semantic smoothing. Similar to the run of agglomerative clustering, we test eleven translation coefficients (\( \lambda \)) ranging from 0 to 1 for semantic smoothing. For standard k-means, we use traditional vector cosine as similarity measure on three different representation schemes: TF, normalized TF, and TF-IDF. Since the result of K-Means clustering varies with the initialization. We run ten times with random initialization and take the average as the result. During the comparative experiment, each run has the same initialization.

4.2. Datasets and indexing schemes

We do clustering experiments on three datasets: TDT2, LA Times (from TREC), and 20-newsgroups (20NG). The TDT2 corpus has 100 document classes, each of which reports a major news event. LA Times news are labeled with 21 unique section names, e.g., Financial, Entertainment, Sports, etc. 20-Newsgroups dataset is collected from 20 different Usenet newsgroups, 1,000 articles from each. We selected 7,094 documents in TDT2 that have a unique class label, 18,547 documents from top ten sections of LA Times, and all 19,997 documents in 20-newsgroups.
The ten classes selected from TDT2 are 20001, 20015, 20002, 20013, 20070, 20044, 20076, 20071, 20012, and 20023. The ten sections selected from LA Times are Entertainment, Financial, Foreign, Late Final, Letters, Metro, National, Sports, Calendar, and View. All 20 classes of 20NG are selected for testing.

<table>
<thead>
<tr>
<th>Dataset Name</th>
<th>TDT2</th>
<th>LA Times</th>
<th>20NG</th>
</tr>
</thead>
<tbody>
<tr>
<td># of indexed docs</td>
<td>7094</td>
<td>18547</td>
<td>19997</td>
</tr>
<tr>
<td># of phrases</td>
<td>8256</td>
<td>9517</td>
<td>10902</td>
</tr>
<tr>
<td>Avg. doc length</td>
<td>240</td>
<td>103</td>
<td>193</td>
</tr>
<tr>
<td># of classes</td>
<td>10</td>
<td>10</td>
<td>20</td>
</tr>
</tbody>
</table>

For each document, we identify individual words and multiword phrases from its title and body. The other sections of a document including Meta data are ignored. For all phrases whose document frequency in the corpus is equal or greater than 10, we will estimate its translation model using the method described in section 2.2. As we stated in the section of introduction, removing stop words may make significant difference for many clustering approaches. However, for a new domain, the choice of stop words is not trivial. The use of inappropriate stop words will even hurt the result. The model-based approaches with semantic smoothing can minimize the impact of stop words. To test this effect, we provide two versions of document indexing, with and without stop words. In this paper, the results not removing stop words are marked with an asterisk. Similarly, many reported results [12] [18] are based on feature selection. A frequently used feature selection approach is to filter out infrequency words. In the experiment, we intentionally keep all words because suitable smoothing can automatically weaken the effect of infrequent words.

We hypothesize that the effect of semantic smoothing on small dataset is stronger than on large dataset because small dataset has serious data sparsity problem when using model-based K-Means. To test this hypothesis, we run partitional clustering on both large and small datasets. A large dataset contains all the documents from selected classes. To build small datasets, we randomly pick 100 random documents from each selected class of a given dataset and then merge them into a big pool for clustering. For each dataset, we create five small datasets and average the experiment results. We test agglomerative clustering on only small datasets because it suffers from the $O(n^2)$ clustering time. For all experiments, we cluster data into $k$ (=the number of selected classes) clusters. Answer key lists (1 large and 5 small subsets of each dataset), and all additional experiment results not shown in this paper are provided when requested.

### 4.3. Agglomerative clustering results

The NMI result of the agglomerative hierarchical clustering using complete linkage criterion is listed in Table 3. When using the vector cosine for pairwise document similarity measure, the TF-IDF scheme performs slightly better than the TF scheme. As we discussed before, the heuristic TF-IDF weighting scheme can discount “general” words and strengthen “specific” words in a document vector. Thus, it can improve the agglomerative clustering quality. The KL-divergence similarity measure combined with background smoothing of document models (i.e., $\lambda=0$ in equation 2) consistently outperforms the cosine measure on both Normalized TF and TF-IDF schemes. As expected, the KL-divergence measure with context-sensitive semantic smoothing significantly improves the quality of the agglomerative clustering on all three datasets. The semantic smoothing not only weakens the effect of class-independent general words, but also assigns reasonable probability to unseen words by phrase-word translations. Since the direct comparison of two documents suffers from the severe data sparsity problem, the semantic smoothing can dramatically improve the quality of agglomerative clustering.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Vector Cosine</th>
<th>KL-Divergence</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TF</td>
<td>TF-IDF</td>
</tr>
<tr>
<td>TDT2</td>
<td>0.369</td>
<td>0.484</td>
</tr>
<tr>
<td>TDT2*</td>
<td>0.141</td>
<td>0.597</td>
</tr>
<tr>
<td>20NG</td>
<td>0.135</td>
<td>0.135</td>
</tr>
<tr>
<td>20NG*</td>
<td>0.099</td>
<td>0.099</td>
</tr>
<tr>
<td>LA Times</td>
<td>0.059</td>
<td>0.054</td>
</tr>
<tr>
<td>LA Times*</td>
<td>0.066</td>
<td>0.114</td>
</tr>
</tbody>
</table>

To see the robustness of the semantic smoothing effect, we show the performance curve in Figure 4&5. Experiments without using stop words are marked with asterisk. Except for the point of $\lambda=1$, the semantic smoothing always improve the cluster quality over the simple background smoothing ($\lambda=0$). In general, NMI will increase with the increase of translation coefficient till the peak point (around 0.7 in our case) and then go downward. In our experiment, we only consider phrases appearing in more than ten documents as topic signatures in order to obtain a good estimate of translation probabilities. Moreover, not all topics in a document can be expressed by multiword phrases. Thus, the phrase-based semantic smoothing will cause information loss and we
interpolate the translation model with a simple language model to make up the loss. Now it is easy to understand why the NMI goes downward when the influence of the semantic smoothing is too high. Actually, LSI also causes information loss when the dimensionality reduction is too aggressive; but there is no mechanism to recover the loss. In this sense, our semantic smoothing approach is better than LSI.

Table 4. NMI results of partitional clustering on small data sets. “Lap”, “Bkg”, and “Semantic” denote Laplacian smoothing, background smoothing, and semantic smoothing, respectively. * means stop words are not removed.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Standard K-Means</th>
<th>Model-based K-Means</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TF</td>
<td>NTF</td>
</tr>
<tr>
<td>TDT2</td>
<td>0.792</td>
<td>0.805</td>
</tr>
<tr>
<td>TDT2*</td>
<td>0.447</td>
<td>0.433</td>
</tr>
<tr>
<td>20NG</td>
<td>0.197</td>
<td>0.161</td>
</tr>
<tr>
<td>20NG*</td>
<td>0.127</td>
<td>0.105</td>
</tr>
<tr>
<td>LATimes</td>
<td>0.194</td>
<td>0.197</td>
</tr>
<tr>
<td>LATimes*</td>
<td>0.128</td>
<td>0.095</td>
</tr>
</tbody>
</table>

Table 5. NMI results of partitional clustering on large data sets. “Lap”, “Bkg”, and “Semantic” denote Laplacian smoothing, background smoothing, and semantic smoothing, respectively. * means stop words are not removed.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Standard K-Means</th>
<th>Model-based K-Means</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TF</td>
<td>NTF</td>
</tr>
<tr>
<td>TDT2</td>
<td>0.685</td>
<td>0.699</td>
</tr>
<tr>
<td>TDT2*</td>
<td>0.479</td>
<td>0.440</td>
</tr>
<tr>
<td>20NG</td>
<td>0.188</td>
<td>0.176</td>
</tr>
<tr>
<td>20NG*</td>
<td>0.075</td>
<td>0.075</td>
</tr>
<tr>
<td>LATimes</td>
<td>0.193</td>
<td>0.203</td>
</tr>
<tr>
<td>LATimes*</td>
<td>0.111</td>
<td>0.061</td>
</tr>
</tbody>
</table>

The comparisons of the semantic smoothing and the background smoothing show different patterns on small datasets and large datasets. As shown in Table 4 & 5, the semantic smoothing performs consistently better than the background smoothing on small datasets whereas the results of two smoothing methods are almost same on large datasets. This is because when the dataset is small, the cluster models still suffers from the data sparsity problem more or less. As we discussed earlier, the semantic smoothing is very effective in solving the data sparsity problem and thus it can significantly improves the clustering results. But when the size of the dataset increases, the data sparsity problem disappears and the semantic smoothing can not take the advantage over the background smoothing any more. This pattern can be observed apparently in Figure 6 & 7. On small datasets, the performance curve will go up with the increase of the translation coefficient (i.e., the influence of semantic smoothing) at first. However, the performance curve on large datasets is almost parallel with the x-axis except the point of λ=1. In other words, the semantic smoothing has no effect on large datasets at all.

The clustering results of partitional clustering on small datasets and large datasets are listed in Table 4 and Table 5, respectively. Of three smoothing schemes for model-based K-Means, Laplacian smoothing always performs the worst on both large datasets and small datasets. This indicates that assigning an equal probability to unseen words is a bad smoothing scheme. It just technically prevents zero probabilities, but can not weaken the effect of general words as the background smoothing does.

4.4. Partitional clustering results

The clustering results of partitional clustering on small datasets and large datasets are listed in Table 4 and Table 5, respectively. Of three smoothing schemes for model-based K-Means, Laplacian smoothing always performs the worst on both large datasets and small datasets. This indicates that assigning an equal probability to unseen words is a bad smoothing scheme. It just technically prevents zero probabilities, but can not weaken the effect of general words as the background smoothing does.

From table 3, we can see stop words have much less impact on TF-IDF, background smoothing and semantic smoothing than on TF because the former approaches automatically weaken the effect of general words on clustering. In cases of TDT2 and LA Times, stop words even seriously hurt the result of TF*IDF.
Similarly to the case of agglomerative clustering, stop words have much less impact on semantic smoothing, background smoothing and TF-IDF than on TF, normalized TF and Laplacian smoothing in context of partitional clustering because the former has the “built-in” mechanism to weaken the effect of general words on clustering. However, it is not trivial to find out a list of stop words for a new domain. The use of improper stop words may even hurt the results. To this sense, the model-based approach with semantic smoothing or background smoothing and the similarity based approach with TF*IDF scheme are always a safe choice for clustering because they produce good result no matter stop words are applied or not.

It’s worth noting that the agglomerative clustering with TF*IDF scheme performs significantly worse than with semantic smoothing whereas the partitional clustering with TF*IDF scheme is consistently as good as or slightly better than with semantic smoothing. We believe this difference is caused by the following two things. First, as mentioned before, the major problem with agglomerative clustering is data sparsity and semantic smoothing is very effective in solving data sparsity problem. Second, the data sparsity issue in partitional clustering becomes minor because a cluster often contains much more than one document and instead, the density of general words becomes the major issue. Although semantic smoothing also has the effect of “discounting” general words, TF*IDF scheme does much more aggressively than semantic smoothing.

5. Conclusions

In this paper, we argue that the sparsity of class-specific “core” words and the density of class-free “general” words are the two major problems with both agglomerative and partitional clustering approaches. We further argue that both problems can be resolved by using suitable model smoothing method. However, to the best of our knowledge, there is no extensive research work studying the relationship between model smoothing methods and clustering quality thus far. The common Laplacian smoothing is developed to avoid zero probability, but not to optimize the model. The background smoothing and the TF*IDF scheme have the effect of “discounting” general words. Background smoothing is helpful to relieve data sparsity, but in a shallow manner. Inspired by a series of statistical translation language models in IR, we propose a novel semantic smoothing method for text clustering use. The basic idea of the method is to identify a set of context-sensitive topic signatures (e.g. multiword phrases) in a document or a cluster and then map topic signatures into individual terms. Thus, an “unseen” core words will be assigned a reasonable probability if its related topic signatures occur in the document or cluster. This statistical translation model is further interpolated with the background collection model. Therefore, the semantic smoothing method can solve both identified problems.

We evaluate the effects of semantic smoothing with a model-based agglomerative clustering and a model-based partitional clustering on three different datasets. The experiment shows that our semantic smoothing is very promising for model-based text clustering. In detail, we obtain following interesting findings from the experiment by comparing semantic smoothing to other five schemes: Laplacian smoothing, background smoothing, TF-cosine, NTF-cosine, and TF*IDF-cosine: (1) Semantic smoothing is much more effective than other schemes on agglomerative clustering where data sparsity is the major problem. (2) The effectiveness of semantic smoothing with partitional clustering depends on the size of the dataset. When dataset is small and data sparsity is the major problem, semantic smoothing is very effective;
otherwise, it equals to background smoothing. (3) Although both semantic smoothing and background smoothing can weaken the effect of general words, they are less effective than TF*IDF which is more aggressive on discounting general words. (4) Laplacian smoothing is the worst among all tested schemes. (5) Stop words have almost no effect on TF*IDF-cosine, background smoothing, and semantic smoothing, but have positive effects on Laplacian smoothing, TF-cosine, and NTF-cosine.

Topic signatures in our semantic smoothing scheme are similar to the latent dimensions in latent semantic indexing (LSI), but in an explicit manner. Because topic signatures are unambiguous in most cases, the translation to individual words is very specific whereas LSI brings noise when exploring latent semantic structures due to term polysemy. LSI also causes information loss during dimensionality reduction. Our approach can recover the loss by interpolating the topic signature translation model with a simple language model. But how to obtain optimal weights for each component in the mixture model is an open problem yet. In addition, we use natural language processing techniques to identify topic signatures from texts. This is somehow ad hoc nature and could be further improved in future.

6. Acknowledgement

This work is supported in part by NSF Career grant (NSF IIS 0448023), NSF CCF 0514679, PA Dept of Health Tobacco Settlement Formula Grant (No. 240205 and No. 240196), and PA Dept of Health Grant (No. 239667).

References
[13] Yoo I., Hu X., Song I-Y, Integration of Semantic-based Bipartite Graph Representation and Mutual Refinement Strategy for Biomedical Literature Clustering, accepted in the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.